In [ ]:
```
# Chris Straschewski
# CMS180014
# CS 4395.001

import sklearn
import csv
import pandas as pd
import tensorflow as tf
import keras
from tensorflow.keras import datasets, layers, models, preprocessing

max_features = 10000
maxlen = 500
batch_size = 32

# data = x
# labels = y
# load dataset
df = pd.read_csv('tripadvisor_hotel_reviews.csv', header=0, usecols=[0,1],

df.replace(1, 0, inplace=True)
df.replace(2, 0, inplace=True)
df.replace(3, 1, inplace=True)
df.replace(4, 1, inplace=True)
df.replace(5, 1, inplace=True)

print('rows and columns:', df.shape)
print(df.head())

# Text Preprocessing
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import TfidfVectorizer

stopwords = set(stopwords.words('english'))
vectorizer = TfidfVectorizer(stop_words=list(stopwords))

# set up X and y
X = df.Review
Y = df.Rating
Y = Y.astype('int')

# take a peek at X
print(X.head())

print(Y[:10])

# split data
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, t

print(X_train.shape)

# apply tfidf vectorizer
X_train = vectorizer.fit_transform(X_train)  # fit and transform the train
X_test = vectorizer.transform(X_test)        # transform only the test dat

print(X_train.shape)
```

```
# CNN
model = models.Sequential()
model.add(layers.Embedding(max_features, 128, input_length=maxlen))
model.add(layers.Conv1D(32, 7, activation='relu'))
model.add(layers.MaxPooling1D(5))
model.add(layers.Conv1D(32, 7, activation='relu'))
model.add(layers.GlobalMaxPooling1D())
model.add(layers.Dense(1))

model.summary()


# compile

model.compile(optimizer=tf.keras.optimizers.RMSprop(lr=1e-4),  # set learn
              loss='binary_crossentropy',
              metrics=['accuracy'])


# train

history = model.fit(X_train,
                    Y_train,
                    epochs=10,
                    batch_size=128,
                    validation_split=0.2)

from sklearn.metrics import classification_report

pred = model.predict(X_test)
pred = [1.0 if p>= 0.5 else 0.0 for p in pred]
print(classification_report(Y_test, pred))

# RNN
# build a Sequential model with Embedding and SimpleRNN layers

model_2 = models.Sequential()
model_2.add(layers.Embedding(max_features, 32))
model_2.add(layers.SimpleRNN(32))
model_2.add(layers.Dense(1, activation='sigmoid'))

model_2.summary()

# compile
model_2.compile(optimizer='rmsprop',
              loss='binary_crossentropy',
              metrics=['accuracy'])

history_2 = model_2.fit(X_train,
                    Y_train,
                    epochs=10,
                    batch_size=128,
                    validation_split=0.2)

pred_2 = model_2.predict(X_test)
pred_2 = [1.0 if p>= 0.5 else 0.0 for p in pred_2]
```

```
print(classification_report(Y_test, pred_2))

# GRU

model_3 = models.Sequential()
model_3.add(layers.Embedding(max_features, 32))
model_3.add(layers.GRU(32))
model_3.add(layers.Dense(1, activation='sigmoid'))

# compile
model_3.compile(optimizer='rmsprop',
              loss='binary_crossentropy',
              metrics=['accuracy'])

# train
history_3 = model_3.fit(X_train,
                    Y_train,
                    epochs=10,
                    batch_size=128,
                    validation_split=0.2)

pred_3 = model_3.predict(X_test)
pred_3 = [1.0 if p>= 0.5 else 0.0 for p in pred_3]
print(classification_report(Y_test, pred_3))
```